International Journal of Foundations of Computer Science
Vol. 30, Nos. 6 & 7 (2019) 1157–1175

DOI: 10.1142/S0129054119400331

Concise Representations of Reversible Automata*

Giovanna J. Lavado† and Luca Prigioniero‡
Dipartimento di Informatica, Università degli Studi di Milano
via Comelico, 39/41 Milan, 20135, Italy
†*lavado@di.unimi.it*
‡*prigioniero@di.unimi.it*

Received 16 November 2017
Accepted 8 June 2018
Communicated by C. Câmpeanu and G. Pighizzini

In this paper, we present two concise representations of reversible automata. Both representations have a size comparable to the size of the minimum equivalent deterministic automaton and can be exponentially smaller than the size of the explicit representations of corresponding reversible automata. Using these representations it is possible to simulate the computations of reversible automata without explicitly writing down their complete descriptions.

1. Introduction

Reversibility is a fundamental principle in physics: in thermodynamics a transformation is reversible if, after occurring, it can be inverted in order to recover the original state of the system. In the study of computational models, reversibility means that each elementary step can be inverted, thus recovering the previous state of the system. In other words, every configuration must admit at most one predecessor. As shown by Landauer, the irreversibility in computation leads to heat dissipations [10], while Toffoli proved that it is ideally possible to build sequential circuits with zero internal power dissipation [16]. This observation suggested to study reversible computations in which there is no loss of information.

Reversibility has been studied on various computational models. In the case of general devices as Turing machines, Bennet proved that each machine can be simulated by a reversible one [1], while Lange, McKenzie, and Tapp proved that each deterministic machine can be simulated by a reversible machine which uses the same amount of space [11]. As a corollary, in the case of a constant amount of

*This paper is an extended version of [14].
‡Corresponding author.

space, this implies that each regular language is accepted by a *reversible two-way deterministic finite automaton*. Actually, this result was already proved by Kondacs and Watrous [6]. In the case of *one-way* automata, the situation is different.[a] The class of languages accepted by *reversible automata* is a proper subclass of the class of regular languages. For example, the regular language a^*b^* cannot be accepted by any reversible automaton [15], even if multiple initial states are allowed. Classical automata, namely automata with a single initial state and a set of final states, have been considered in the works by Holzer, Jakobi, and Kutrib [4, 7, 8]. In particular, they gave a characterization of regular languages which are accepted by reversible automata [4]. This characterization is given in terms of the structure of the minimum deterministic automaton, i.e., the smallest deterministic automaton accepting the language under consideration. Furthermore, the authors provided an algorithm that, in the case the language is acceptable by a reversible automaton, allows to transform the minimum deterministic automaton into an equivalent reversible automaton, which in the worst case is exponentially larger than the given minimum automaton. In spite of that, the resulting automaton is minimal, namely there are no reversible automata accepting the same language with a smaller number of states. However, it is not necessarily unique. In fact, contrary to the minimum deterministic automaton that is unique for a fixed language, there could exist different reversible automata with the same (minimal) number of states accepting the same language. Further results concerning minimality and reducibility for reversible automata have been proved in Refs. [3, 12].

Due to the above mentioned exponential state gap between deterministic automata and equivalent reversible automata, an explicit representation of a minimal reversible automaton can be exponentially larger than the representation of the corresponding minimum deterministic automaton. However, the minimal reversible automaton produced by the construction provided by Holzer, Jakobi, and Kutrib [4] is obtained by creating copies of some parts of the minimum automaton. So, its transition table contains repeated patterns. Thus, it is interesting to investigate whether it is possible to obtain a concise representation of it, by avoiding to repeat those patterns. To this aim, in this paper we present two concise representations of reversible automata, which can be used to simulate reversible computations without explicitly writing down the description of the reversible automaton.

The first representation is based on a parameter β which is equal to the maximum number of incoming transitions with the same letter in each state of the given deterministic automaton A. Given β and A it is possible to simulate the computations of a reversible automaton A' equivalent to A, without explicitly representing it. The drawback of this simple representation is that even when the given

[a]From now on, we will consider only *one-way automata*. Hence we will omit to specify "one-way" all the times.

automaton A is minimum, the simulated reversible automaton A' is not necessarily minimal. This motivates us to look for a different concise representation, which exploits the fact that all the minimal reversible automata accepting a language have the same "state structure", in the sense that, for each given A, there exists a function $c : Q \to \mathbf{N}$, such that for each state q in A, there exist exactly $c(q)$ states equivalent to q in each minimal reversible deterministic automaton equivalent to A [12]. The second representation is given in terms of the minimum deterministic automaton A accepting the language under consideration and such function c. We prove that, by using such representation, it is possible to simulate the behaviour of a minimal reversible automaton equivalent to A without explicitly representing it. Both representations have polynomial size with respect to the size of the given deterministic automaton A and require a precomputation (of the parameter β and of the function c, respectively) which can be performed in linear time.

2. Preliminaries

In this section, we recall some basic definitions and results that will be used in the paper. For a detailed exposition, we refer the reader to [5]. Given a set S, let us denote by $\#S$ its cardinality and by 2^S the family of all of its subsets. Given an alphabet Σ, $|w|$ denotes the length of a string $w \in \Sigma^*$ and ε the empty string.

A *deterministic automaton* is a tuple $A = (Q, \Sigma, \delta, q_I, F)$, where Q is the set of *states*, Σ is the *input alphabet*, $q_I \in Q$ is the *initial state*, $F \subseteq Q$ is the set of *accepting states*, and $\delta : Q \times \Sigma \to Q$ is the partial *transition function*. The function δ can be extended to words in the standard way. The *language accepted* by A is $L(A) = \{w \in \Sigma^* \mid \delta(q_I, w) \in F\}$. The *reverse* transition function of A is the function $\delta^R : Q \times \Sigma \to 2^Q$, with $\delta^R(p, a) = \{q \in Q \mid \delta(q, a) = p\}$. A state $p \in Q$ is *useful* if p is *reachable*, i.e., there exists $w \in \Sigma^*$ such that $\delta(q_I, w) = p$, and *productive*, i.e., if there is $w \in \Sigma^*$ such that $\delta(p, w) \in F$. When the set of states Q is finite, the automaton A is said to be a deterministic *finite* automaton (DFA). In this paper we only consider automata with only useful states.

We say that two states $p, q \in Q$ are *equivalent* if for all $w \in \Sigma^*$, $\delta(p, w) \in F$ exactly when $\delta(q, w) \in F$. Two automata A and A' are said to be *equivalent* if they accept the same language, i.e., $L(A) = L(A')$. By *minimal automaton* (in a certain family of automata) we mean an automaton with a minimal number of states. When the minimal automaton is unique (e.g., for the family of all DFAs accepting a certain regular language) we call it *minimum*.

A *strongly connected component* (SCC) C of a DFA $A = (Q, \Sigma, \delta, q_I, F)$ is a maximal subset C of Q such that in the transition graph of A there exists a path between each pair of states in C. We introduce the relation $\prec$ on the set of SCCs of A, such that, for two such components C_1 and C_2, $C_1 \prec C_2$ when no state in C_1 can be reached from a state in C_2, but a state in C_2 is reachable from a state in C_1. As usual, if $C_1 \prec C_2$ or $C_1 = C_2$ we write $C_1 \preceq C_2$. It can be verified that $\preceq$ is a partial order.

Given a DFA $A = (Q, \Sigma, \delta, q_I, F)$, a state $r \in Q$ is said to be *irreversible* when $\#\delta^R(r, a) \geq 2$ for some $a \in \Sigma$. In this case, for each $q \in \delta^R(r, a)$, the transition from q on a is said to be *irreversible*. If a state is not irreversible, then it is said to be *reversible*. The DFA A is said to be *irreversible* if it contains at least one irreversible state, otherwise A is *reversible* (REV-DFA). As pointed out in Ref. [8], the notion of reversibility for a language is related to the computational model under consideration. In this paper we only consider DFAs. Hence, by saying that a language L is *reversible*, we refer to this model, namely we mean that there exists a REV-DFA accepting L. The following result presents a characterization of reversible languages:

Theorem 1. [4, Theorem 2] *Let L be a regular language and $M = (Q, \Sigma, \delta, q_I, F)$ be the minimum* DFA *accepting L. Then, L is accepted by a* REV-DFA *if and only if there do not exist two states $p, q \in Q$, a letter $a \in \Sigma$, and a string $w \in \Sigma^*$ such that $p \neq q$, $\delta(p, a) = \delta(q, a)$, and $\delta(q, aw) = q$.*

According to Theorem 1, a language L is reversible exactly when the minimum DFA accepting it does not contain the *forbidden pattern* consisting of two transitions on the same letter a entering in the same state r, with one of these transitions arriving from a state in the same SCC as r. An algorithm to convert a minimum DFA M into an equivalent REV-DFA, if any, was obtained in Ref. [4]. Furthermore, the resulting REV-DFA is minimal.

We present an outline of the algorithm. It iteratively builds a REV-DFA A in the following way. At the beginning A is a copy of M. Then, the algorithm considers a minimal (with respect to $\preceq$) SCC C that contains an irreversible state and replaces it with a number of copies which is equal to the maximum number of transitions on the same letter incoming in a state of C. This process is iterated until all the states in A are reversible.

In Ref. [4], it has also been observed that there are reversible languages having several nonisomorphic minimal REV-DFAs, while in [12, Lemmata 5 and 6] the following result has been proved:

Lemma 2. *Let $M = (Q, \Sigma, \delta, q_I, F)$ be the minimum* DFA *accepting a reversible language L. There exists a function $c : Q \to \boldsymbol{N}$ such that for each state $q \in Q$:*

- *in any* REV-DFA *equivalent to M there are at least $c(q)$ different states equivalent to q.*
- *in any minimal* REV-DFA *equivalent to M there are exactly $c(q)$ different states equivalent to q.*

Furthermore, if $p, q \in Q$ are in the same SCC*, then $c(p) = c(q)$.*

3. A Simple Concise Representation

In this section we present our first concise representation. Let us start with a construction for simulating a DFA by an equivalent REV-DFA, in which we use the

information about the maximum number of incoming transitions with respect to the same letter in the irreversible states.

Let $A = (Q, \Sigma, \delta, q_I, F)$ be a DFA with all useful states and let β be the maximum number of transitions on the same letter incoming in a state of A, i.e.,

$$\beta = \max\{\#\delta^R(q, a) \mid q \in Q, a \in \Sigma\}.$$

We observe that $\beta > 1$ if and only if A is irreversible. We define the following automaton with infinitely many states $A_\infty = (Q \times \mathbf{N}, \Sigma, \delta', q'_I, F \times \mathbf{N})$ where:

- $q'_I = \langle q_I, 0\rangle$,
- The transitions are defined as follows:
  Let $\delta(q, a) = p$ and $\delta^R(p, a) = \{q_{j_1}, \ldots, q_{j_k}\}$, $k \geq 1$ for $q, p \in Q$, $a \in \Sigma$. Then, for $x \geq 0$:

$$\delta'(\langle q, x\rangle, a) = \begin{cases} \langle p, x\rangle & \text{if } k = 1 \\ \langle p, x\beta + i - 1\rangle & \text{otherwise} \end{cases} \tag{1}$$

  where $i \in \{1, \ldots, k\}$ is such that $q = q_{j_i}$.

Notice that, if $\delta'(\langle q, x\rangle, a) = \langle p, y\rangle$ then $x \leq y$. Roughly speaking, the idea of the construction is to use the second component of the states of A_∞ as labels in order to distinguish different copies of a state reached from an irreversible transition in A. The formula used for the second component allow us to obtain this goal, as we will prove in Theorem 3.

We denote by A' the automaton obtained by restricting A_∞ to useful states.

Theorem 3. *Let $A = (Q, \Sigma, \delta, q_I, F)$ be a* DFA *and $A' = (Q', \Sigma, \delta', q'_I, F')$ be the automaton obtained by applying the above construction to A, restricted to useful states. Then $L(A') = L(A)$ and A' is reversible.*

Proof. First, it is enough to observe that each state $\langle q, x\rangle \in Q'$ is equivalent to $q \in Q$.

Second, we have to prove that for each $a \in \Sigma$, $\langle \bar{q}_1, x_1\rangle, \langle \bar{q}_2, x_2\rangle \in Q'$, $\langle \bar{q}_1, x_1\rangle \neq \langle \bar{q}_2, x_2\rangle$, implies that if both $\delta'(\langle \bar{q}_1, x_1\rangle, a)$ and $\delta'(\langle \bar{q}_2, x_2\rangle, a)$ are defined then they are different. Observe that $\delta'(\langle \bar{q}_i, x_i\rangle, a)$, for $i \in \{1, 2\}$, can be undefined only if $\delta(\bar{q}_i, a)$ is undefined. We consider the following cases:

- If $\bar{q}_1 = \bar{q}_2$ and $x_1 \neq x_2$ then $\delta(\bar{q}_1, a) = \delta(\bar{q}_2, a) = p$ for some $p \in Q$, otherwise M would be nondeterministic. Let $\delta^R(p, a) = \{q_{j_1}, \ldots, q_{j_k}\}$, $k \geq 1$. Then there exists an i such that $\bar{q}_1 = \bar{q}_2 = q_{j_i}$. Considering (1), $\delta'(\langle \bar{q}_1, x_1\rangle, a) = \langle p, y_1\rangle$ and $\delta'(\langle \bar{q}_1, x_2\rangle, a) = \langle p, y_2\rangle$.
  - If $k = 1$ then $y_1 = x_1$ and $y_2 = x_2$.
  - If $k > 1$ then $y_1 = x_1\beta + i - 1$ and $y_2 = x_2\beta + i - 1$.

  Since $x_1 \neq x_2$ we get $y_1 \neq y_2$. Hence, $\langle p, y_1\rangle \neq \langle p, y_2\rangle$.

- If $\bar{q}_1 \neq \bar{q}_2$ and $\delta(\bar{q}_1, a) = p_1 \neq \delta(\bar{q}_2, a) = p_2$, then in A' the states $\delta'(\langle \bar{q}_1, x_1\rangle, a) = \langle p_1, y_1\rangle$ and $\delta'(\langle \bar{q}_2, x_2\rangle, a) = \langle p_2, y_2\rangle$ are different regardless of the values of y_1 and y_2.
- If $\bar{q}_1 \neq \bar{q}_2$ and $\delta(\bar{q}_1, a) = \delta(\bar{q}_2, a) = p$, let $\delta^R(p, a) = \{q_{j_1}, \ldots, q_{j_k}\}$, with $k > 1$. Then, there exist i, i', with $i \neq i'$ such that $\bar{q}_1 = q_{j_i}$ and $\bar{q}_2 = q_{j_{i'}}$. Considering (1), $\delta'(\langle \bar{q}_1, x_1\rangle, a) = \langle p, y_1\rangle$ and $\delta'(\langle \bar{q}_2, x_2\rangle, a) = \langle p, y_2\rangle$, where $y_1 = x_1\beta + i - 1$ and $y_2 = x_2\beta + i' - 1$. In the case of $x_1 = x_2$, since $i \neq i'$, we get $y_1 \neq y_2$. In the case of $x_1 \neq x_2$, and supposing, without loss of generality, $x_1 > x_2$, we get $x_1\beta \geq x_2\beta + \beta$, and hence $x_1\beta > x_2\beta + \beta - 1 \geq x_2\beta + i' - 1$ (notice that $i' \leq \beta$). Then $y_1 = x_1\beta + i - 1 \geq x_1\beta > x_2\beta + i' - 1 = y_2$. This implies that $\langle p, y_1\rangle \neq \langle p, y_2\rangle$.

Hence, $\delta'(\langle \bar{q}_1, x_1\rangle, a) \neq \delta'(\langle \bar{q}_2, x_2\rangle, a)$. This allows us to conclude that A' is reversible. □

We showed that A' simulates A. We have now to prove that it is finite if and only if A does not contain the forbidden pattern.

Theorem 4. *The automaton $A' = (Q', \Sigma, \delta, q'_I, F')$ obtained by applying the above construction to a* DFA *$A = (Q, \Sigma, \delta, q_I, F)$ is infinite if and only if A contains the forbidden pattern.*

Proof. ($\Rightarrow$) If A' contains infinitely many states then for each $m > 0$ there exists a string $w \in \Sigma^*$ such that $\delta'(q'_I, w) = \langle q, x\rangle$ with $x \geq m$ and $\langle q, x\rangle \in Q'$ is an useful state. Without loss of generality we can choose w in such a way that the second component of the state which is reached in A' by reading any proper prefix of w is smaller than x. Thus $w = w_1 a_1 w_2 a_2 \cdots w_\ell a_\ell$, with $w_1, \ldots, w_\ell \in \Sigma^*$, $a_1, \ldots, a_\ell \in \Sigma$, such that irreversible transitions are used on $a_{\hat{\ell}}$ only, for $\hat{\ell} = 1, \ldots, \ell$.

We consider the path on w in A'. By construction, the second component of a state is changed only while reading symbols $a_{\hat{\ell}}$, $\hat{\ell} = 1, \ldots, \ell$: let $\delta'(\langle q_I, 0\rangle, w_1) = \langle p_1, x_0\rangle$, $\delta'(\langle p_1, x_0\rangle, a_1) = \langle p'_1, x_1\rangle$, $\delta'(\langle p'_1, x_1\rangle, w_2) = \langle p_2, x_1\rangle$, $\delta'(\langle p_2, x_1\rangle, a_2) = \langle p'_2, x_2\rangle, \ldots, \delta'(\langle p_\ell, x_{\ell-1}\rangle, a_\ell) = \langle p'_\ell, x_\ell\rangle$. Then, $0 = x_0 < x_1 < x_2 < \cdots < x_{\ell-1} < x_\ell$. For $\ell > \#Q \cdot \#\Sigma$ we can find two values $i \neq j$, such that the pairs (a_i, p'_i) and (a_j, p'_j) are equal. Without loss of generality suppose $i < j$. Then in A we have $\#\delta^R(p'_i, a_i) > 1$ and $\delta(p'_i, w_{i+1}a_{i+1}, \ldots, w_j a_j) = p'_j = p'_i$, which gives the forbidden pattern in A.

($\Leftarrow$) If A contains the forbidden pattern then there exist two useful states $p_1, p_2 \in Q$, $a \in \Sigma$, $w \in \Sigma^*$ such that $p_1 \neq p_2$, $\delta(p_1, a) = \delta(p_2, a)$ and $\delta(p_2, aw) = p_2$. Let $r = \delta(p_1, a)$ and $\delta^R(r, a) = \{q_{j_1}, \ldots, q_{j_k}\}$. Since $p_1 \neq p_2$ we get $k > 1$ and two different indices i, i' such that $p_1 = q_{j_i}$ and $p_2 = q_{j_{i'}}$. Moreover, $\beta > 1$. Now we want to prove that the simulation of the forbidden pattern bring us to have infinitely many useful states in A'.

Due to $p_1 \neq p_2$, in A' there exist two different states of the form $\langle p_1, x_1\rangle$ and $\langle p_2, x_2\rangle$ such that $\delta'(\langle p_1, x_1\rangle, a) = \langle r, x_1\beta + i - 1\rangle$ and

$\delta'(\langle p_2, x_2\rangle, a) = \langle r, x_2\beta + i' - 1\rangle$. Let $\hat{x}_2 = x_2\beta + i' - 1$. Since from Theorem 3 A' is reversible, we get $x_1\beta + i - 1 \neq \hat{x}_2$.

Since r and p_2 belong to the same SCC, in A' there exists a path from $\langle r, \hat{x}_2\rangle$ to some state $\langle p_2, y_2\rangle$ with $y_2 \geq \hat{x}_2$. Due to $\delta(p_2, a) = r$, there is a transition $\delta'(\langle p_2, y_2\rangle, a) = \langle r, y_2\beta + i' - 1\rangle$. Let $\hat{x}_3 = y_2\beta + i' - 1$. Since $\beta > 1$ we get $\hat{x}_3 > y_2 \geq \hat{x}_2$. Thus, we have $\hat{x}_3 > \hat{x}_2$. This implies $\langle r, \hat{x}_2\rangle \neq \langle r, \hat{x}_3\rangle$. We can iterate this argument by obtaining a sequence of infinitely many useful states $\langle r, \hat{x}_\ell\rangle$, $\ell \geq 0$ and $\hat{x}_{\ell-1} < \hat{x}_\ell$. This allows us to conclude that A' contains infinitely many states. □

Two examples related to the previous construction are shown in Figs. 1 and 2, where $\beta = 2$. Let us apply the construction to transform the DFA shown in Fig. 1 through an equivalent REV-DFA. Given for instance $\delta(q_3, b) = q_5$, we have $\delta^R(q_5, b) = \{q_3, q_4\}$, $k > 1$ and $i = 1$. Then $\delta'(\langle q_3, 1\rangle, b) = \langle q_5, 2\rangle$.

Now we apply the same construction to the DFA in Fig. 2. Given for instance $\delta(q_1, b) = q_2$, we have $\delta^R(q_2, b) = \{q_I, q_1\}$, $k > 1$ and $i = 2$. Then $\delta'(\langle q_1, 0\rangle, b) = \langle q_2, 1\rangle$. This time taking $\delta(q_2, a) = q_3$, we have $\delta^R(q_3, a) = \{q_1, q_2\}$, $k > 1$ and $i = 2$. Then $\delta'(\langle q_2, 1\rangle, a) = \langle q_3, 3\rangle$. Actually, the simulation of a computation on a string does not require the explicit construction of the automaton A'. In fact, once we have β the computation of the automaton can be obtained,

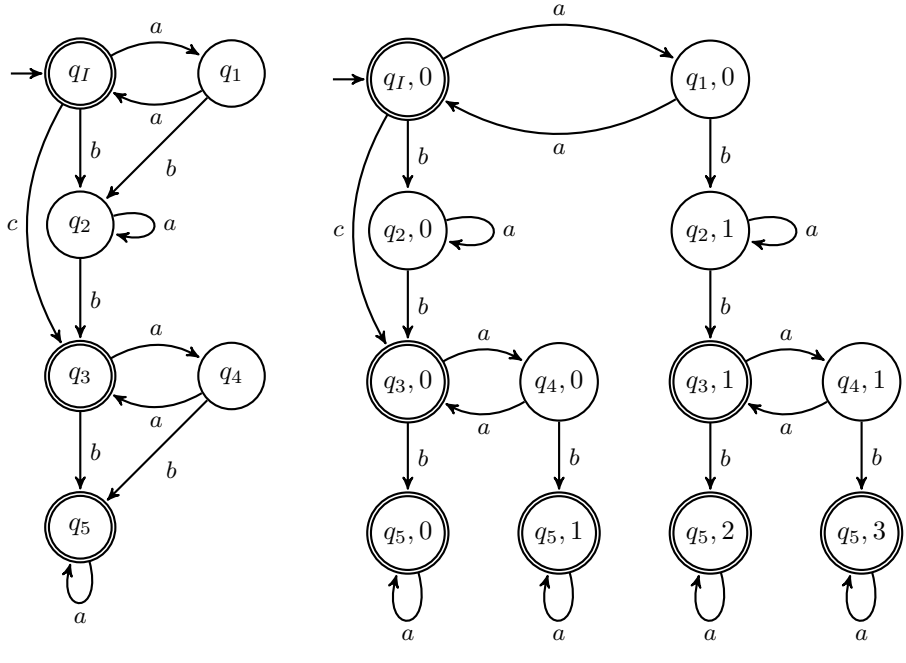

Fig. 1. A DFA where $\beta = 2$ and the equivalent REV-DFA obtained by using the construction.

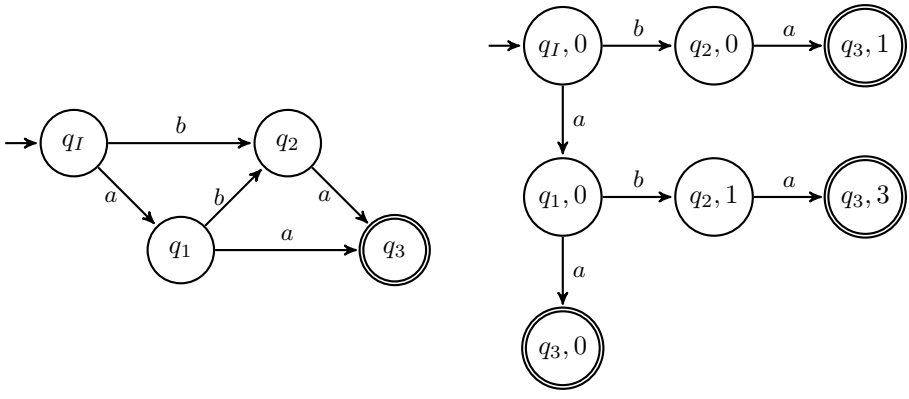

Fig. 2. A DFA where $\beta = 2$ and the equivalent REV-DFA obtained by using the construction.

using the transition table of A and (1). For instance on aba we have the following steps: $q'_I \xrightarrow{a} \langle q_1, 0\rangle \xrightarrow{b} \langle q_2, 1\rangle \xrightarrow{a} \langle q_3, 3\rangle$.

Notice that the second components in the states having the same q are not necessarily consecutive numbers, in the sense that it is possible to have some gaps in the numbering as illustrated in Fig. 2 (states of the form $\langle q_3, x\rangle$ in the automaton on the right).

We point out that the automaton A' can be simulated without explicitly constructing its transition table. Indeed, to simulate A' it is enough to know the value of β, which can be computed from the transition table of A, and to follow the transitions of A applying (1) to compute the states reached by A'. So, a concise representation of A' is given by the value of β and the automaton A. We will discuss later in this section how to compute β and how much the value of the second component of a state of A' can be large.

Even when applied to a minimum DFA, the above construction produces a REV-DFA which is not necessarily minimal as illustrated in Figs. 3 and 4: in Fig. 3 a minimum DFA $M = (Q, \Sigma, \delta, q_I, F)$ and an equivalent minimal REV-DFA (obtained by applying the algorithm in Ref. [4]) are shown. Notice that the minimal REV-DFA contains five states which are equivalent to q_7. Instead, Fig. 4 shows the REV-DFA A' obtained by the above construction (notice that $\beta = 3$ and the chosen order for each set of states with outgoing transitions to a reversible state with the same letter is: (q_2, q_3) for $\delta^R(q_4, b)$, (q_I, q_5, q_6) for $\delta^R(q_7, a)$, and (q_5, q_6) for $\delta^R(q_7, b)$). In particular, A' contains six states equivalent to q_7.

It could be interesting to observe that, by choosing a different order for the states in $\delta^R(q, a)$, $q \in Q$, $a \in \Sigma$, and applying the same construction, the minimum automaton in Fig. 3 is simulated by a minimal REV-DFA. For example, the minimal REV-DFA in Fig. 5 is obtained by applying the construction to A and considering the following order: (q_3, q_2) for $\delta^R(q_4, b)$, (q_6, q_5, q_I) for $\delta^R(q_7, a)$, and (q_6, q_5) for $\delta^R(q_7, b)$.

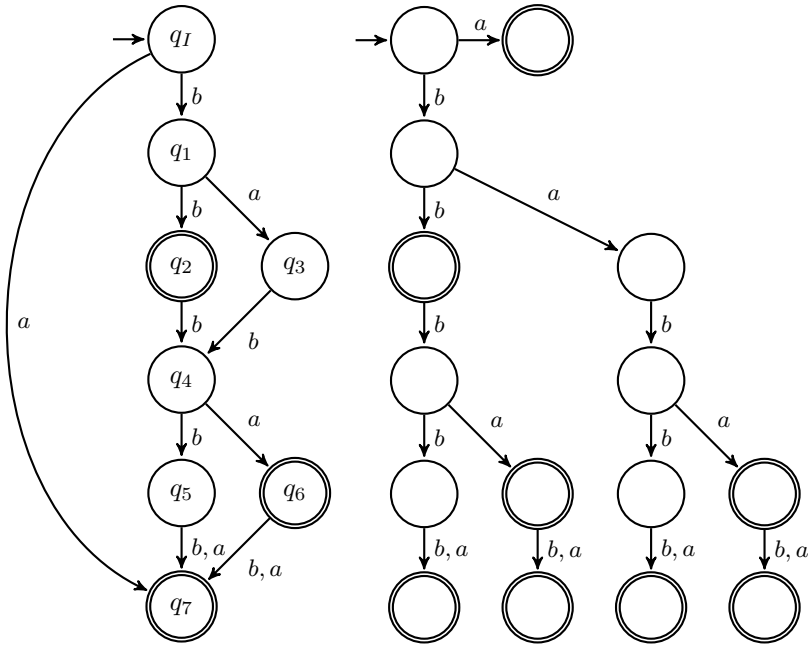

Fig. 3. A minimum DFA and an equivalent minimal REV-DFA.

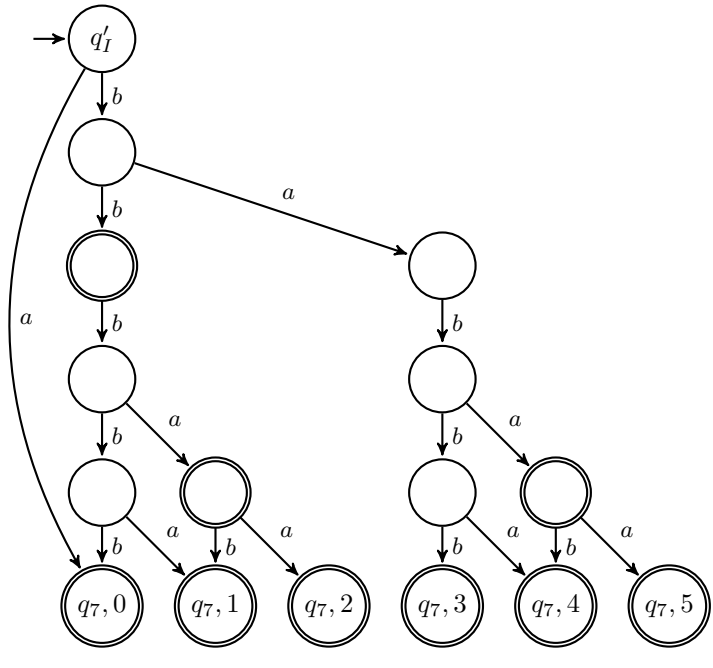

Fig. 4. A non minimal REV-DFA obtained by applying the construction to the minimum DFA in Fig. 3 where $\beta = 3$.

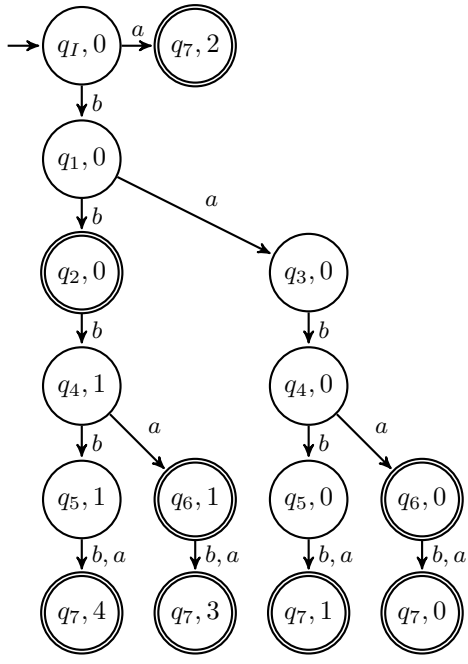

Fig. 5. A minimal REV-DFA obtained by applying the construction to the DFA in Fig. 3, for a suitable chosen order for the states with outgoing transitions on the same symbol to irreversible states.

There also exist cases in which the (non)minimality is not influenced by the selected order of the states. Let us consider, for example, the automaton in Fig. 6. Recalling that the simulation increases the value of x in each pair $\langle q, x\rangle$ only when an irreversible transition is encountered along a path, let us show how the construction works. For the sake of simplicity, we proceed by analyzing the states in topological order.

Starting from $q'_I = \langle q_I, 0\rangle$ and reading $b(a + b)$, no irreversible state is passed through, so the second parts of the state pairs do not change. Reading the symbol b, the state q_4 can be reached from q_2 and q_3. Since there are no copies of these states, the only possible reached states should be $\langle q_2, 0\rangle$ and $\langle q_3, 0\rangle$, leading to the state set $\{\langle q_4, 0\rangle, \langle q_4, 1\rangle\}$, independently from the chosen order of the states $\{q_2, q_3\} = \delta^R(q_4, b)$. Thus, by reading a or b, it is possible to reach the states $\{\langle q_5, 0\rangle, \langle q_6, 0\rangle, \langle q_5, 1\rangle, \langle q_6, 1\rangle\}$. Let us now consider the state q_7: it can be reached from $\langle q_I, 0\rangle$, $\langle q_2, 0\rangle$, and $\langle q_3, 0\rangle$ with the letter a, thus leading to the states in $\{\langle q_7, 0\rangle, \langle q_7, 1\rangle, \langle q_7, 2\rangle\} = r_{q_7,a}$, independently from the chosen order of the elements in $\delta^R(q_7, a)$. Furthermore, q_7 can be also reached from the states q_5 and q_6 by reading the symbol b. Recalling that the only reached states in the simulation equivalent to q_5 and q_6 are $\langle q_5, 0\rangle$, $\langle q_6, 0\rangle$, $\langle q_5, 1\rangle$, and $\langle q_6, 1\rangle$, it is possible to reach the states in $\{\langle q_7, 0\rangle, \langle q_7, 1\rangle, \langle q_7, 3\rangle, \langle q_7, 4\rangle\} = r_{q_7,b}$ by reading a b, independently

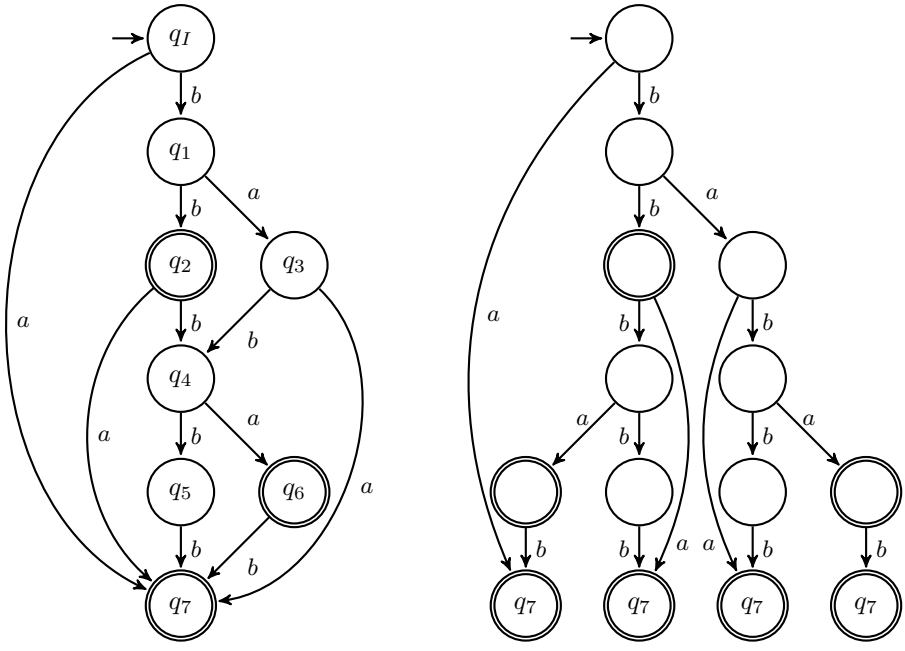

Fig. 6. A minimum DFA for which it is not possible to simulate the REV-DFA by using the construction based on the knowledge of β and an equivalent minimal REV-DFA obtained by applying the algorithm outlined in Sec. 2.

from the chosen order of the states in $\delta^R(q_7, b)$. So, considering all the reached states of the simulated automaton, it is possible to observe that there are five copies equivalent to q_7, while the minimum number of copies of such a state, obtainable by applying the algorithm by Holzer, Jakobi, and Kutrib outlined in Sec. 2, would be four, as shown in Fig. 6, i.e., $\#(r_{q_7,a} \cup r_{q_7,b}) > c(q_7)$, where $c(q)$ denotes the number of different states equivalent to q in any minimal REV-DFA (cf. Lemma 2).

In Theorem 4 it has been proved that when a DFA A does not contain the forbidden pattern, then the automaton A' obtained by applying the above construction is finite. Furthermore, by Theorem 3, A' is reversible and, as already observed, not necessarily minimal. Hence, it is interesting to know what is the maximum value of the second component in a state of A'. In order to give a bound we will use the following lemmata.

Lemma 5. *Given a* DFA A *with at least two states, if the initial state* q_I *is not reversible or every state other than* q_I *is not reversible, then* A *contains the forbidden pattern.*

Proof. Let $A = (Q, \Sigma, \delta, q_I, F)$ be a DFA. Since transitions entering the initial state q_I can only arrive from states in the same SCC of q_I, if the initial state would be irreversible, then A should contain the forbidden pattern.

Suppose now that q_I is reversible and every state other than q_I is irreversible. Let p_0 be one of them. Then, there are two incoming transitions in p_0 with the same letter a_0 from two different states. At most one of them could be q_I, while the other one, say p_1, must be irreversible. Hence, even in p_1 there are two incoming transitions with the same letter a_1 from two different states, where at least one of them, say p_2, is irreversible. By iterating the argument we can construct an arbitrarily long sequence of irreversible states $p_0, p_1, p_2, \ldots$ and letters $a_0, a_1, a_2, \ldots$ such that $\#\delta^R(p_\ell, a_\ell) > 1$ and $\delta(p_{\ell+1}, a_\ell) = p_\ell$ for $\ell \geq 0$. Since Q is finite, sooner or later we will find i, j, $i < j$, such that $p_j = p_i$. This implies that there is a string $w = a_{j-1}a_{j-2}\cdots a_{i+1}$ such that $\delta(p_j, wa_i) = p_i$. Since $p_i = p_j$ and p_i has two incoming transitions on the same letter a_i we conclude that A contains the forbidden pattern. □

We are now ready to prove the following lemma.

Lemma 6. *Let $A' = (Q', \Sigma, \delta', q'_I, F')$ be the automaton obtained by applying the above construction to a* DFA *$A = (Q, \Sigma, \delta, q_I, F)$ which does not contain the forbidden pattern. Given $w \in \Sigma^*$ and $q = \delta(q_I, w)$, let $p_1, \ldots, p_k \in Q$ be all the states reached by irreversible transitions during the computation of A on input w. Then $\delta'(q'_I, w) = \langle q, x\rangle$, where $0 \leq x < \beta^k$.*

Proof. Let $w_0, \ldots, w_k$ be $k+1$ strings such that $w = w_0 \cdots w_k$ and $\delta(q_I, w_0) = p_1, \delta(p_1, w_1) = p_2, \ldots, \delta(p_{k-1}, w_{k-1}) = p_k, \delta(p_k, w_k) = q$. Notice that if q is an irreversible state, then $p_k = q$ and $w_k = \varepsilon$.

By construction of A', there exists the path $\delta'(\langle q_I, 0\rangle, w_0) = \langle p_1, x_1\rangle$, $\delta'(\langle p_1, x_1\rangle, w_1) = \langle p_2, x_2\rangle, \ldots, \delta'(\langle p_{k-1}, x_{k-1}\rangle, w_{k-1}) = \langle p_k, x_k\rangle$, $\delta'(\langle p_k, x_k\rangle, w_k) = \langle q, x\rangle$. It is possible to observe that, since p_k is the last irreversible state along the path from q_I to q, $x_k = x$. Then, from the definition of δ', we have $0 \leq x_1 < \beta$, $0 \leq x_2 = x_1\beta < \beta^2$, $0 \leq x_3 < \beta^3, \ldots, 0 \leq x_{k-1} < \beta^{k-1}$, $0 \leq x_k = x < \beta^k$. □

As a consequence of Lemma 6, the value of the second components of states of A' is smaller than β^k, where k is the maximum possible number of irreversible states along a path starting in the initial state. Considering Lemma 5, we obtain:

Corollary 7. *If a* DFA *A with $\#Q \geq 2$ does not contain the forbidden pattern, then the maximum value of the second component of a state of A' obtained applying the above construction to A is smaller than $\beta^{\#Q-2}$.*

Observe that the maximum value of the second component in a state of A' is reached when in each irreversible state r of A the maximum number of incoming transitions for the same letter a is equal to β, i.e., $\#\delta^R(r, a) = \beta$. Two examples have been shown in Figs. 1 and 2. The DFA on the left of Fig. 2 has a path from q_I to q_3 reading the string $w = aba$ containing all irreversible states.

We also observe that β has an important role in the construction, so we believe useful to outline how β can be computed (Algorithm 1). Given a

Algorithm 1. Computation of β from a DFA $A = (Q, \Sigma, \delta, q_I, F)$ where $Q = \{q_1, \ldots, q_n\}$

```
1: Let C be an array of size n
2: β ← 0
3: for all a ∈ Σ do
4:     for i ← 1 to n do
5:         C[q_i] ← 0
6:     for i ← 1 to n do
7:         q_t ← δ(q_i, a)
8:         C[q_t] ← C[q_t] + 1
9:     m ← max{C[q_i] | q_i ∈ Q}
10:    if m ≥ β then β ← m
11: return β
```


DFA $A = (Q, \Sigma, \delta, q_I, F)$ containing only useful states, we assume that δ resides in a transition table T of size $\#Q \cdot \#\Sigma$. The key observation is that a state is irreversible with respect to a symbol when it occurs more than one time in a column of T. Hence, the problem can be reduced to find the maximum number of occurrences of a state in a column of T. The algorithm uses an array in which the number of occurrences of each state is stored (Lines 4–8). For each column of T we can find the maximum number of occurrences of some state (Line 9). The value can be stored in a variable and updated at each iteration. In this way the algorithm keeps in the variable the current maximum value. The final value of the variable is the value of β. Since we need to visit each column of T, the overall time is $\mathcal{O}(\#Q \cdot \#\Sigma)$, which is linear in the cardinality of Q.

4. Another Concise Representation

The drawback of the representation described in Sec. 3 is that the reversible automaton is not necessarily minimal. In this section we give a different representation which avoids such problem. Before proceeding, let us recall the properties related to minimal REV-DFAs shown in Lemma 2. According to these, all the minimal REV-DFAs accepting L have the same "state structure", in the sense that all of them should contain exactly $c(q)$ states equivalent to each state q of M.

Here we present an easy way to compute the value of $c(q)$, for each $q \in Q$ of a given automaton M not containing the forbidden pattern, that is summarized in Algorithm 2.

The algorithm firstly transforms the transition graph of M by decomposing it in SCCs, replacing each SCC by a single state, and linking with an edge two SCCs $\mathcal{C}'$ and $\mathcal{C}''$, with $\mathcal{C}' \neq \mathcal{C}''$, if there exist two states $p \in \mathcal{C}'$ and $q \in \mathcal{C}''$ such that $\delta(p, a) = q$ for some symbol a. This is summarized in Line 1. After that, the obtained acyclic

Algorithm 2. Computation of $c(p)$ for each $p \in Q$.

1: Let $\mathcal{S}_M$ be the graph representing the SCCs of the transition graph of M
2: Let $\mathcal{L}_{\mathcal{S}_M}$ be the list of the SCCs of M sorted by topological order $\preceq$
3: **for all** SCCs $\mathcal{C} \in \mathcal{L}_{\mathcal{S}_M}$ **do**
4: max_c $\leftarrow 1$
5: **for all** states $q \in \mathcal{C}$ **do**
6: **for all** letters $a \in \Sigma$ **do**
7: max_c $\leftarrow \max\{\text{max\_c}, \sum_{p \in \delta^R(q,a)\setminus\mathcal{C}} c(p)\}$
8: **for all** states $q \in \mathcal{C}$ **do**
9: $c(q) \leftarrow$ max_c
10: **return** c

graph $\mathcal{S}_M$ can be sorted in topological order[b] ($\preceq$, Line 2). For further details about these constructions see, for example, [2, Chapter 23].

Then, all $c(q)$ are computed by analyzing the SCCs in topological order in the following way (Lines 3–9): when a SCC $\mathcal{C}$ is considered, first of all the algorithm computes for each state $q \in \mathcal{C}$ the maximum number of transitions on the same symbol a entering q from SCCs different from $\mathcal{C}$, where a transition from p to q is *counted* $c(p)$ times, i.e., the algorithm computes $\sum_{p \in \delta^R(q,a)\setminus\mathcal{C}} c(p)$, for all $q \in Q$ and $a \in \Sigma$ and stores the maximum of all such values (Lines 5 – 7). This value is assigned as $c(q)$ to each $q \in \mathcal{C}$ (Lines 8–9). Note that, analyzing the SCCs in topological order, the value of $c(p)$ is used for all the states p in the set $\delta^R(q,a)\setminus\mathcal{C}$ when the algorithm is going to compute $c(q)$, for $q \in \mathcal{C}$. Obviously, for each state q in the first SCC C_{q_I}, $\delta^R(q,a)\setminus\mathcal{C}_{q_I} = \emptyset$.

If M does not contain the forbidden pattern, then for each $p \in \mathcal{C}_{q_I}$, $c(p) = 1$ and the set $\delta^R(p,a)\setminus\mathcal{C}_{q_I}$ is empty. As a consequence, the instruction at Line 7 does not produce any increment of *max_c* for any state in the SCC under consideration.

It is easy to see that Algorithm 2 works in polynomial time: it is well known that operations at Lines 1 and 2 require time $\mathcal{O}(\#V + \#E)$, where V and E are, respectively the set of vertices and the set of edges of the graph under consideration. So, in our case, the time for compute $\mathcal{S}_\mathcal{M}$ and $\mathcal{L}_{\mathcal{S}_\mathcal{M}}$ is $\mathcal{O}(\#Q)$. From Line 3 to 9, the algorithm analyzes the incoming transitions to each state q. This can be done in time $\mathcal{O}(\#Q)$ assuming that Σ is fixed. So, the Algorithm 2 uses $\mathcal{O}(\#Q)$ time.

The following property will be useful for the construction:

Lemma 8. *Let* $\delta^R(p,a) = \{q_{j_1}, \ldots, q_{j_k}\}$, $k \geq 1$, $p, q_{j_1}, \ldots, q_{j_k} \in Q$, *and* $a \in \Sigma$. *Then* $\sum_{h=1}^{k} c(q_{j_h}) < c(p)$ *and, consequently,* $\sum_{h=1}^{i-1} c(q_{j_h}) + x < c(p)$, *for* $i = 1, \ldots, k$, $0 \leq x < c(q_{j_i})$.

[b]Notice that the first SCC, according to the topological order, is the one containing the initial state. Hence, there are no edges from other SCCs entering it.

Proof. We observe that the elements of the set $\{\sum_{h=1}^{i-1} c(q_{j_h}) + x \mid 0 \leq x < c(q_{j_i})\}$, entirely cover an interval $[0, m)$ with $m = \sum_{h=1}^{k} c(q_{j_h})$.

We are going to prove that $m \leq c(p)$.

From the Algorithm 2, $m = \sum_{h=1}^{k} c(q_{j_h}) = \sum_{q \in \delta^R(p,a) \setminus \mathcal{C}} c(q)$ is the number of transitions on symbol a entering p when all the components $\mathcal{C}' \preceq \mathcal{C}$, with $\mathcal{C}' \neq \mathcal{C}$, have been processed (and the algorithm is going to consider the component $\mathcal{C}$).

Since Algorithm 2 computes $c(p)$ for each state $p \in \mathcal{C}_p$ by selecting the maximum number of transitions on the same letter entering a state in $\mathcal{C}_p$, we can conclude that $m \leq c(p)$. □

We are now ready to present the construction which leads to our second concise representation. Let $M = (Q, \Sigma, \delta, q_I, F)$ be a minimum DFA accepting a reversible language L. We define the following DFA $A' = (Q', \Sigma, \delta', q'_I, F')$:

- $Q' = \{\langle q, x\rangle \mid q \in Q, 0 \leq x < c(q)\}$,
- $q'_I = \langle q_I, 0\rangle$,
- $F' = \{\langle q, x\rangle \mid q \in F, 0 \leq x < c(q)\}$,
- The transitions are defined as follows:
  Let $\delta(q, a) = p$ and $\delta^R(p, a) = \{q_{j_1}, \ldots, q_{j_k}\}$, $k \geq 1$ for $q, p \in Q$, $a \in \Sigma$. Then:

$$\delta'(\langle q, x\rangle, a) = \left\langle p, \sum_{h<i} c(q_{j_h}) + x \right\rangle, \tag{2}$$

  where $i \in \{1, \ldots, k\}$ is such that $q = q_{j_i}$ and $0 \leq x < c(q)$.
  Notice that by Lemma 8 the second component is less than $c(q)$. Hence the function δ' is well defined.

We will prove that A' is a minimal REV-DFA equivalent to M.

Two examples related to the construction are shown in Figs. 7 and 8. Let us apply the construction to the minimum DFA M in Fig. 2, which gives the machine shown

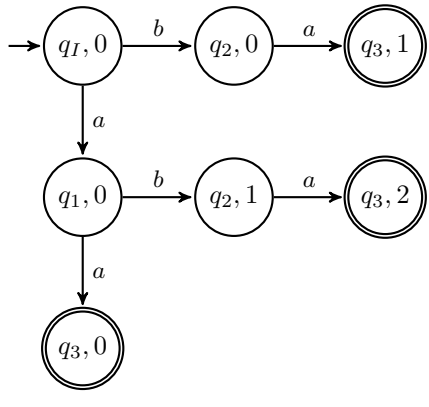

Fig. 7. A minimal REV-DFA obtained by applying the construction based on the knowledge of $c(q)$ for each $q \in Q$ to the minimum DFA in Fig. 2.

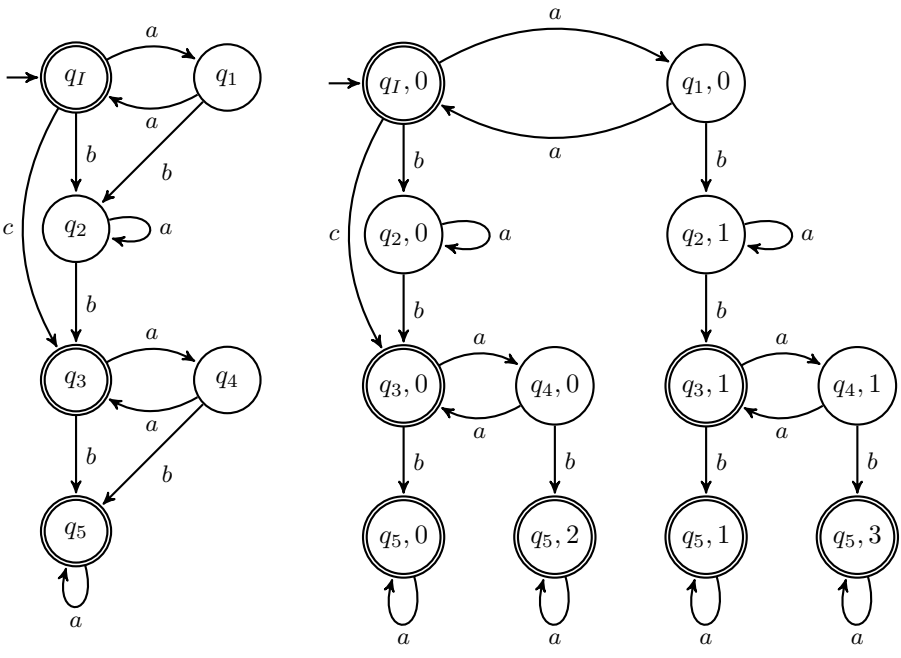

Fig. 8. A DFA and an equivalent minimal REV-DFA.

in Fig. 7. The topological order $\preceq$ of the SCCs of M clearly is $q_I \preceq q_1 \preceq q_2 \preceq q_3$.[c] In particular, $c(q_I) = c(q_1) = 1$, $c(q_2) = 2$, $c(q_3) = 3$. Given for instance $\delta(q_1, b) = q_2$, $\delta^R(q_2, b) = \{q_I, q_1\}$, then $\delta'(\langle q_1, 0\rangle, b) = \langle q_2, c(q_I) + 0\rangle = \langle q_2, 1\rangle$.

Notice that, by applying the construction of Sec. 3 to the same DFA (Fig. 2), there are differences between the second components of the states equivalent to q_3. In particular, in the previous construction, the set of the second components of equivalent states could contain gaps, even if the resulting automaton was minimal. By applying the construction described in this section, instead, the set of the second components of the states equivalent to any state $q \in A$ is equal to $\{0, \ldots, c(q) - 1\}$. Furthermore, we remind the reader that the reversible automaton obtained by applying the construction described in Sec. 3 is not necessarily minimal.

Now we apply the construction to the minimum DFA M in Fig. 8. The states of M are grouped into the SCCs $\{q_I, q_1\}, \{q_2\}, \{q_3, q_4\}, \{q_5\}$. Consider the following number of copies $c(q)$: $c(q_I) = c(q_1) = 1$, $c(q_2) = c(q_3) = c(q_4) = 2$ and $c(q_5) = 4$. For instance, given $\delta(q_4, b) = q_5$, $\delta^R(q_5, b) = \{q_3, q_4\}$, we have $\delta'(\langle q_4, 0\rangle, b) = \langle q_5, c(q_3) + 0\rangle = \langle q_5, 2\rangle$.

[c]A generalization of this example in which the number of copies $c(q)$ of a state $q \in Q$ follows the sequence of Fibonacci is given in [4, Example 9].

Note that, even in this case, it is possible to simulate an irreversible computation of the REV-DFA A' without explicitly constructing it: given a letter and knowing the state in which the automata is, given by the tuple $\langle state, index \rangle$, it is always possible to obtain the next state. As example, consider the minimal DFA showed in Fig. 8 (on the left) and the input string $x = abbab$. So, the computation of the simulated REV-DFA passes through the following states: $\langle q_I, 0\rangle \xrightarrow{a} \langle q_1, 0\rangle \xrightarrow{b} \langle q_2, 0+1\rangle = \langle q_2, 1\rangle \xrightarrow{b} \langle q_3, 1+0\rangle = \langle q_3, 1\rangle \xrightarrow{a} \langle q_4, 1\rangle \xrightarrow{b} \langle q_5, 1+2\rangle = \langle q_5, 3\rangle$.

The main difference between this simulation and the previous one based on the knowledge of β is that, starting from the minimum DFA accepting a reversible language, the resulting automaton is minimal. Such a property is proved in the following theorem.

Theorem 9. *Let $M = (Q, \Sigma, \delta, q_I, F)$ be a minimum* DFA *accepting a reversible language L. Let $A' = (Q', \Sigma, \delta', q'_I, F')$ be the* DFA *obtained by applying the construction to M, then $L(A') = L$, and A' is reversible and minimal.*

Proof. First, it is enough to observe that each $\langle q, x\rangle \in Q'$ is equivalent to $q \in Q$.

Second, we have to prove that for each $a \in \Sigma$, $\langle \bar{q}_1, x_1\rangle, \langle \bar{q}_2, x_2\rangle \in Q'$, $\langle \bar{q}_1, x_1\rangle \neq \langle \bar{q}_2, x_2\rangle$, implies that if both $\delta'(\langle \bar{q}_1, x_1\rangle, a)$ and $\delta'(\langle \bar{q}_2, x_2\rangle, a)$ are defined then they are different. Observe that $\delta'(\langle \bar{q}_i, x_i\rangle, a)$, for $i \in \{1, 2\}$, can be undefined only if $\delta(\bar{q}_i, a)$ is undefined. We proceed by studying the following cases:

- If $\bar{q}_1 = \bar{q}_2$ and $x_1 \neq x_2$ then $\delta(\bar{q}_1, a) = \delta(\bar{q}_2, a) = p$ for some $p \in Q$, otherwise M would be nondeterministic. Let $\delta^R(p, a) = \{q_{j_1}, \ldots, q_{j_k}\}$, $k \geq 1$. Then there exists i such that $\bar{q}_1 = \bar{q}_2 = q_{j_i}$. Considering (2), we have $\delta'(\langle \bar{q}_1, x_1\rangle, a) = \langle p, y_1\rangle$ and $\delta'(\langle \bar{q}_1, x_2\rangle, a) = \langle p, y_2\rangle$, where $y_1 = \sum_{h<i} c(q_{j_h}) + x_1$ and $y_2 = \sum_{h<i} c(q_{j_h}) + x_2$. Since $x_1 \neq x_2$ we conclude that $y_1 \neq y_2$, hence $\langle p, y_1\rangle \neq \langle p, y_2\rangle$.
- If $\bar{q}_1 \neq \bar{q}_2$ then either $\delta(\bar{q}_1, a) = p_1$ and $\delta(\bar{q}_2, a) = p_2$, with $p_1 \neq p_2$ or $\delta(\bar{q}_1, a) = \delta(\bar{q}_2, a) = p$.
  - If $p_1 \neq p_2$, then the states $\delta'(\langle \bar{q}_1, x_1\rangle, a) = \langle p_1, y_1\rangle$ and $\delta'(\langle \bar{q}_2, x_2\rangle, a) = \langle p_2, y_2\rangle$ in A' are different regardless of the values of y_1 and y_2.
  - If $\delta(\bar{q}_1, a) = \delta(\bar{q}_2, a) = p$, then let $\delta^R(p, a) = \{q_{j_1}, \ldots, q_{j_k}\}$, $k > 1$. Since $\bar{q}_1 \neq \bar{q}_2$, there exist i, i', with $i \neq i'$ such that $\bar{q}_1 = q_{j_i}$ and $\bar{q}_2 = q_{j_{i'}}$. Hence, $\delta'(\langle \bar{q}_1, x_1\rangle, a) = \langle p, y_1\rangle$ and $\delta'(\langle \bar{q}_2, x_2\rangle, a) = \langle p, y_2\rangle$, where $y_1 = \sum_{h<i} c(q_{j_h}) + x_1$ and $y_2 = \sum_{h<i'} c(q_{j_h}) + x_2$, $0 \leq x_1 < c(q_{j_i})$ and $0 \leq x_2 < c(q_{j_{i'}})$. Without loss of generality, suppose $i < i'$. Hence, $y_1 = \sum_{h<i} c(q_{j_h}) + x_1 < \sum_{h \leq i} c(q_{j_h}) \leq \sum_{h<i'} c(q_{j_h}) \leq y_2$. This implies that $\langle p, y_1\rangle \neq \langle p, y_2\rangle$.

Third, we observe that by Lemma 8, A' contains at most $c(p)$ copies of any state $p \in Q$. However since A' is reversible, by Lemma 2 it should contain at least $c(p)$ many different states equivalent to p. Hence we conclude that A' is a REV-DFA containing exactly $c(p)$ copies of each state p of the minimum DFA M. According to Lemma 2 this implies that A' is minimal. □

According to the results in this section, given a minimum DFA M, after computing $c(q)$ for each state q of M, we can simulate a minimal REV-DFA A' equivalent to M, without explicitly representing it, starting from the initial state $q'_I = \langle q_I, 0\rangle$ and using (2) at each step to compute the next state. Since A' can have exponentially many states with respect to M, this avoids to write down a large description.

5. Conclusion

We have presented two concise representations of a reversible automaton A' equivalent to a given DFA A. Both of them allow to simulate the REV-DFA without explicitly writing down its transition table which, in the worst case, can be exponentially larger. In particular, the algorithm for converting a DFA accepting a reversible language to an equivalent minimal REV-DFA given in Ref. [4] iteratively creates copies of SCCs of A in order to eliminate irreversibility. Here, for the simulation of A', we follow the computation on A and we use a variable, updated at each step of computation, containing the copy index of the corresponding SCC in A'.

The first representation in Sec. 3 requires an easy precomputation of a parameter β, but the simulated automaton is not necessarily minimal. Instead, the second representation in Sec. 4 requires the more involved precomputation of the function c, but the simulated automaton is minimal. Both precomputations can be done in polynomial time. Notice that the time required for the simulation of one computation step of a REV-DFA in one of these forms is the same as for the given DFA, with additional constant time for updating the variable.

Even when the REV-DFA A' obtained from a minimum DFA A in the first representation is not minimal, its size is not too far from the size of a minimal REV-DFA in the following sense. In Lemma 6 we gave an upper bound of the maximum value of the second component in a state of A', i.e., β^{k_p}, where k_p is the maximum number of irreversible states on a path in A from the initial state q_I to p. Since A' is reversible we have $c(p) \leq \beta^{k_p}$ (Lemma 2). Furthermore, in the path at least two copies of each irreversible state should be created to obtain a reversible automaton. Then, $2^{k_p} \leq c(p) \leq \beta^{k_p}$. This implies that A' has a polynomial number of states with respect to the number of states of A if and only if each minimal REV-DFA equivalent to A has a polynomial number of states.

Recently, the definition of reversibility for finite automata has been deepened and relaxed, thus allowing different degrees of reversibility, the degree one corresponding to reversible automata [9]. In particular, finite automata whose computations can be reversed deterministically by knowing the last k symbols read from the input, for a fixed k (*reversed k-lookahead*), have led to the notion of *weak irreversibility* [13]. It could be interesting to extend the results about the succinctness of representations of REV-DFA to weakly irreversible automata.

References

[1] C. Bennett, Logical reversibility of computation, *IBM Journal of Research and Development* **17**(6) (1973) 525–532.

[2] T. H. Cormen, *Introduction to Algorithms* (MIT Press, 2009).

[3] K. Gelle and S. Iván, Reversible languages having finitely many reduced automata, *AFL 2017*, *EPTCS* **252** (2017) 114–127.

[4] M. Holzer, S. Jakobi and M. Kutrib, Minimal reversible deterministic finite automata, *DLT 2015*, *Lecture Notes in Computer Science* **9168** (2015) 276–287.

[5] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages and Computation* (Addison-Wesley, 1979).

[6] A. Kondacs and J. Watrous, On the power of quantum finite state automata, *FOCS* (IEEE Computer Society, 1997), pp. 66–75.

[7] M. Kutrib, Aspects of reversibility for classical automata, *Computing with New Resources*, *Lecture Notes in Computer Science* **8808** (2014) 83–98.

[8] M. Kutrib, Reversible and irreversible computations of deterministic finite-state devices, *MFCS 2015*, *Lecture Notes in Computer Science* **9234** (2015) 38–52.

[9] M. Kutrib and T. Worsch, Degrees of reversibility for DFA and DPDA, *RC 2014. Proceedings*, *Lecture Notes in Computer Science* **8507** (2014) 40–53.

[10] R. Landauer, Irreversibility and heat generation in the computing process, *IBM Journal of Research and Development* **5** (1961) 183–191.

[11] K. Lange, P. McKenzie and A. Tapp, Reversible space equals deterministic space, *J. Comput. Syst. Sci.* **60**(2) (2000) 354–367.

[12] G. J. Lavado, G. Pighizzini and L. Prigioniero, Minimal and reduced reversible automata, *Journal of Automata, Languages and Combinatorics* **22**(1–3) (2017) 145–168.

[13] G. J. Lavado, G. Pighizzini and L. Prigioniero, Weakly and strongly irreversible regular languages, *AFL 2017*, *EPTCS* **252** (2017) 143–156.

[14] G. J. Lavado and L. Prigioniero, Concise representations of reversible automata, *DCFS 2017*, *Lecture Notes in Computer Science* **10316** (2017) 238–249.

[15] J. Pin, On reversible automata, *LATIN '92*, *Lecture Notes in Computer Science* **583** (1992) 401–416.

[16] T. Toffoli, Reversible computing, *Automata, Languages and Programming*, *Lecture Notes in Computer Science* **85** (1980) 632–644.